Presented at the 25th Asilomar Conference on Signals, Systems, and Computers.

A Joint Source/Channel Coder Design*

Fuling Liu and Khalid Sayood
Dept. of Electrical Engineering
and the Center for Communication
& Information Science
Univ. of Nebraska, Lincoln, NE 68588-0511

Jerry D. Gibson†
Information Systems Laboratory
and the Telecommunications Program
Dept. of Electrical Engineering
Stanford Univ., Stanford, CA 94305

Abstract

Source coders and channel coders are generally designed separately without reference to each other. This approach is justified by a famous result of Shannons. However, there are many situations in practice in which the assumptions upon which this result is based are violated. Specifically, we examine the situation where there is residual redundancy at the source coder output. We have previously shown that this residual redundancy can be used to provide error correction using a Viterbi decoder. In this paper we present the second half of the design; the design of encoders for this situation. We show through simulation results that the proposed coders consistently outperform conventional source-channel coder pairs with gains of up to 12dB at high probability of error.

1 Introduction

One of Shannon's many fundamental contributions was his result that source coding and channel coding can be treated separately without any loss of performance for the overall system [1]. The basic design procedure is to select a source encoder which changes the source sequence into *iid* bits followed by a channel encoder which encodes the bits for reliable transmission over the channel. However, the separation argument no longer holds if either of the following two situations occur:

i. The input to the source decoder is different from the output of the source encoder, which happens when the link between the source encoder and source decoder is no longer error free, or

ii. The source coder output contains redundancy.

Case (i) occurs when the channel coder does not achieve zero error probability and case (ii) occurs when the source encoder is suboptimal. These two situations are common occurrences in practical systems where source or channel models are imperfectly known, complexity is a serious issue, or significant delay is not tolerable. Approaches developed for such situations are usually grouped under the general heading of joint source/channel coding.

Most joint source channel coding approaches can be classified in two main categories; (A) approaches which entail the modification of the source coder/decoder structure to reduce the effect of channel errors [2-10], and (B) approaches which examine the distribution of bits between the source and channel coders [11, 12]. The first set of approaches can be divided still further into two classes. One class of approaches examines the modification of the overall structure [2-5], while the other deals with the modification of the decoding procedure to take advantage of the redundancy in the source coder output [6-10].

In this paper we present an approach to joint source/channel coder design, which belongs to category A, and hence we explore a technique for designing joint source/channel coders, rather than ways of distributing bits between source coders and channel coders. We assume that the two nonideal situations referred to earlier are present. For a nonideal source coder, we use MAP arguments to design a decoder which takes advantage of redundancy in the source coder output to perform error correction. We then use the decoder structure to infer the encoder design.

2 The Design Criterion

For a discrete memoryless channel (DMC), let the channel input alphabet be denoted by $A =$

*This work was supported in part by NASA Lewis Research Center (NAG 3-806) and NASA Goddard Space Flight Center (NAG 5-916)

†On leave from Dept. of Electrical Engr. Texas A&M Univ.

$\{a_0, a_1, \ldots, a_{M-1},\}$, and the channel input and output sequences by $Y = \{y_0, y_1, \ldots, y_{L-1}\}$ and $\hat{Y} = \{\hat{y}_0, \hat{y}_1, \ldots, \hat{y}_{L-1}\}$, respectively. If $\mathcal{A} = \{A_i\}$ is the set of sequences $A_i = \{\alpha_{i,0}, \alpha_{i,1}, \ldots, \alpha_{i,L-1}\}$, $\alpha_{i,k} \epsilon A$, then the optimum receiver (in the sense of maximizing the probability of making a correct decision) maximizes $P[C]$, where

$$P[C] = \sum_{A_i} P[C|\hat{Y}]P[\hat{Y}].$$

This in turn implies that the optimum receiver maximizes $P[C|\hat{Y}]$. When the receiver selects the output to be A_k, then $P[C|\hat{Y}] = P[Y = A_k|\hat{Y}]$. Thus, the optimum receiver selects the sequence A_k such that

$$P[Y = A_k|\hat{Y}] \geq P[Y = A_i|\hat{Y}] \ \forall_i.$$

Lemma 1

Let y_i be the input to a DMC. Given y_{i-1}, y_i is conditionally independent of $y_{n-k}, k > 1$. If $\hat{y}_0 = y_0$ then the optimum receiver selects a sequence A_i to maximize $\Pi_{i=1}^{L-1} P(y_i|y_{i-1}, \hat{Y}_i)$ where $\hat{Y}_k = \{\hat{y}_k, \hat{y}_{k+1}, \ldots, \hat{y}_{L-1}\}$.

The lemma addresses the situation in case (ii), i.e., the situation in which the source coder output (which is also the channel input sequence) contains redundancy. Using this lemma, we can design a decoder which will take advantage of dependence in the channel input sequence. The lemma provides the mathematical structure for the decoder. The physical structure can be easily obtained by examining the quantity to be maximized. The optimum decoder maximizes $P(Y|\hat{Y})$ or equivalently $\log P(Y|\hat{Y})$, but

$$\log P(Y|\hat{Y}) = \sum \log P(y_i|\hat{Y}_i, y_{i-1}) \qquad (1)$$

which is similar in form to the path metric of a convolutional decoder. Error correction using convolutional codes is made possible by explicitly limiting the possible codeword to codeword transitions, based on the previous code input and the coder structure. In this case, while there is no structure being imposed by the encoder, there is sufficient residual structure in the source coder output that can be used for error correction. This structure can be quantified in light of the Lemma. That is, the structure is reflected in the conditional probabilities, and can be utilized via the path metric in (1) in a decoder similar in structure to a convolutional decoder. However, to implement this decoder we need to be able to compute the path metric. Unfortunately the quantity $P(y_i|\hat{Y}_i, y_{i-1})$ is difficult to estimate. We have therefore used various approximations to this quantity with some success. In [8, 9] $P(y_i|\hat{Y}_i, y_{i-1})$ is approximated by $P(y_i|\hat{y}_i, y_{i-1})$ with excellent results. Other approximations can be found in [13].

In [9] we showed that the use of the decoder led to dramatic improvements under high error rate conditions. However at low error rates the performance improvement was from nonexistent to minimal. This is in contrast to standard error correcting approaches, in which the greatest performance improvements are at low error rates, with a rapid deterioration in performance at high error rates. In this work we combine the two approaches to develop a joint source channel codec which provides protection equal to the standard channel encoders at low error rates while providing significant error protection at high error rates.

3 Proposed Encoder Structure

In the conventional error protection approach we introduce structure in the transmitted bitstream. In the approach proposed in [9], we use the residual structure in the (generally nonbinary) source coder output sequence. To combine the two approaches, we need to introduce additional structure without disturbing the structure already present. Because of the nature of the decoding approach, a convolutional encoder would be most appropriate for introducing structure. However, a standard binary convolutional encoder will tend to destroy the structure in the source coder output. To preserve the residual structure while introducing additional structure we propose to use nonbinary convolutional encoders (NCE) whose input alphabet is the output alphabet of the source coder.

Let x_n, the input to the NCE, be selected from the alphabet $A = \{0, 1, 2, ..., N-1\}$, and let y_n, the output alphabet of the NCE, be selected from the alphabet $S = \{0, 1, 2, ..., M-1\}$. Then, two of the proposed NCEs can be described by the following mappings

1. $M = N^2$; $y_n = Nx_{n-1} + x_n$

The number of bits required to represent the output alphabet using a fixed length code is

$$\lceil \log_2(M) \rceil = \lceil \log_2(N^2) \rceil = \lceil 2 \log_2(N) \rceil$$

Therefore in terms of rate, this coder is equivalent to a rate 1/2 convolutional encoder. The encoder memory in bits is $2\lceil \log_2(N) \rceil$ as each output value depends on two input values.

As an example, consider the situation when $N = 4$. Then $A = \{0, 1, 2, 3\}$ and $S = \{0, 1, 2, ..., 15\}$. Given

the input sequence x_n : 0 1 3 0 2 1 1 0 3 3 and assuming the encoder is initialized with zeros, the output sequence will be y_n : 0 1 7 12 2 9 5 4 3 15.

The encoder memory is four bits. Notice that while the encoder output alphabet is of size N^2, at any given instant the encoder can only emit one of N different symbols as should be the case for a rate 1/2 convolutional encoder. For example if $y_{n-1} = 0$, then y_n will take on a value from $\{0, 1, 2, ..., (N-1)\}$. In general, given a value for y_{n-1}, y_n will take on a value from $\{\alpha N, \alpha N + 1, \alpha N + 2, ..., \alpha N + N - 1\}$, where $\alpha = y_{n-1}(mod N)$. This structure can be used by the decoder to provide error protection. The encoder is shown in Figure 1a.

2. $M = N^3$; $y_n = N^2 x_{2n-2} + N x_{2n-1} + x_{2n}$

The final encoder we consider is equivalent to a rate 2/3 convolutional coder. Notice that while the input output relationship looks similar to a rate 1/3 encoder, we generate one output for every two inputs. Thus, while the number of bits needed to represent one letter from the output alphabet is three times the bits needed to represent a letter from the input alphabet, because two input letters are represented by a single output letter, the rate is 2/3. Again, assuming a value of 4 for N, the output alphabet is of size 64, and for the input sequence used previously, the output sequence is y_n : 0 52 35 22 49 3.

The encoder memory is again 6 bits. A block diagram of the encoder is shown in Figure 1b. The rate of the encoder can also be inferred from the fact that while the encoder output alphabet is of size N^3, at any instant the encoder can transmit one of N^2 (instead of N) symbols. Given a value for y_{n-1}, y_n can take on a value from the alphabet $\{\gamma N^2, \gamma N^2 + 1, ..., \gamma N^2 + (N^2 - 1)\}$ where $\gamma = y_{n-1}(mod N)$.

4 Binary Encoding of the NCE Output

We will make use of the residual structure in the source coder output (which is preserved in the NCE output) at the receiver. However, we can also make use of this structure in selecting binary codes for the NCE output. An intelligent assignment of binary codes can improve the error correcting performance of the system. Our strategy is to try to maximize the Hamming distance between codewords that are likely to be mistaken for one another.

First we obtain a partition of the alphabet based on the fact that given a particular value for y_{n-1}, y_n can only take on values from a subset of the full alphabet. To see this, consider the rate 1/2 NCE; then the alphabet S can be partitioned into the following sub-alphabets:

$$S_j = (jN, jN + 1, ..., jN + N - 1) \quad j = 0, 1, ...N - 1$$

where the encoder will select letters from alphabet S_j at time n if $j = y_{n-1}(\mathrm{mod} N)$. Now for each sub-alphabet we have to pick N codewords out of M $(= N^2)$ possible choices. We first pick the sub-alphabet containing the most likely letter. The letters in the sub-alphabet are ordered according to their probability of occurrence. We assign a codeword a from the list of available codewords to the most probable symbol. Then, assign the complement of a to the next symbol on the list. Therefore the distance between the two most likely symbols in the list is $K = \lceil \log_2 M \rceil$ bits. We then pick a codeword b from the list which is at a Hamming distance of $K/2$ from a and assign it and its complement to the next two elements on the list. This process is continued with the selection of letters that are$K/2^k$ away from a at the k^{th} step until all letters in the subalphabet have a codeword assigned to them. We then pick the sub-alphabet that contains the next most likely letter. It is assigned the available codeword at maximum distance from a. The procedure for assigning codewords within the sub-alphabet is then repeated. The assignment for a rate 1/2 with $N = 4$ code is shown in Table 1.

5 Simulation Results

The proposed approach was simulated using a two-bit DPCM system as the source coder, and the three NCE described in section 3. The source used were standard test images USC Girl, USC Couple and a 256x256 portion of Lena. The decoder structure used was that of a Viterbi decoder with branch metric $\log L$

$$L = \frac{P(\hat{y}_i \mid y_i)\, P(y_i \mid y_{i-1}, y_{i-2})}{P(\hat{y}_i)}$$

where y_i denotes the NCE output and $\hat{y}_i$ denotes the corrupted channel output. The probabilities $P(y_i \mid y_{i-1}, y_{i-2})$ were estimated using a training sequence. This requires estimating MN^2 probabilities, which were estimated using the USC Girl image. The test images were the USC Couple and Lena images.

The proposed scheme was compared with a conventional source coder-convolutional coder combination. The source coder and source sequence were the same in both systems. The convolutional codes selected were the codes with maximal d_{free} and the

same rate and memory characteristics as the proposed NCEs from [14]. The performance measure was the signal-to-noise-ratio (SNR) defined as

$$SNR = 10 \log_{10} \frac{\sum u_i^2}{\sum (u_i - \hat{u}_i)^2}$$

where u_i is the input to the source encoder and $\hat{u}_i$ is the output of the source decoder.

The results show consistent improvement in performance for the proposed system. At low probabilities of error both systems perform very well. At high probabilities of error ($> 10^{-2}$), however, there is a substantial improvement in performance when the proposed system is used.

In Figures 2a and 2b we show the results of one of the simulations for the rate 1/2 codes. The binary assignment of Table 1 was used in the simulation. Notice the flatness of the performance curve for the proposed system. While the proposed system consistently outperforms the conventional system, it is at higher probabilities of error that the differences really become significant. At a probability of error of 10^{-1} there is almost a 6dB difference in the performance of the two systems! This "flattening out" of the performance curve makes the approach useful for a large variety of channel error conditions.

Similar performance improvements can be seen for the rate 2/3 system of the second mapping. The performance curves are shown in Figure 3. Notice that again the proposed system consistently outperforms the conventional system. In this case at a probability of error of 10^{-1} the performance improvement is more than 12dB! In fact, the proposed rate 2/3 system performs better than the conventional rate 1/2 system.

6 Conclusion

If the source and channel coder are designed in a "joint" manner, that is the design of each takes into account the overall conditions (source as well as channel statistics), we can obtain excellent performance over a wide range of channel conditions. In this paper we have presented one such design. The resulting performance improvement seems to validate this approach.

References

[1] C. E. Shannon. *Bell Syst. Tech. J.* 27:379–423, 623–656. 1948.

[2] E. Ayanoğlu and R. M. Gray. *IEEE Trans. Inform. Theory.* IT-33:855–865. Nov. 1987.

[3] T. C. Ancheta, Jr. Ph.D. dissertation, Dept. of Electrical Engr., Univ. of Notre Dame. Aug. 1977.

[4] K–Y Chang and R. W. Donaldson. *IEEE Trans. Commun.* COM-20:338–350. June 1972.

[5] D. J. Goodman and C. E. Sundberg. *Bell Syst. Tech. J.* 62:2017–203. Sept. 1983.

[6] R. C. Reininger and J. D. Gibson. *IEEE Trans. Commun.* COM-31:572–577. April 1983.

[7] R. Steele et al. *IEEE Trans. Commun.* COM-27:252–255. Jan. 1979.

[8] K. Sayood and J. C. Borkenhagen. *Proceedings IEEE ICC '86.* 1888–1892. June 1986.

[9] K. Sayood and J. C. Borkenhagen. *IEEE Transactions on Communications.* COM-39. June 1991.

[10] K. Sayood and J. D. Gibson. *Proc. 22nd Annual CISS*, Princeton, NJ. 380-385. Mar. 1988.

[11] J. W. Modestino et al. *IEEE Trans. Commun.* COM-29:1262–1274. Sept. 1981.

[12] D. Comstock and J. D. Gibson. *IEEE Trans. Commun.* COM-32:856–861. July 1984.

[13] K. Sayood, J. D. Gibson, and F. Liu. *Proc. 22nd Annual Asilomar Conference on Circuits, Systems, and Computers.* 102-106. Nov. 1988.

[14] S. Lin and D. J. Costello. *Error Control Coding.* Prentice Hall. 1983.

Table 1: Codeword Assignments

Symbol	Code	Symbol	Code
0	0000	8	1011
1	0011	9	0111
2	1100	10	0100
3	1111	11	1000
4	1110	12	0101
5	1101	13	1001
6	0001	14	1010
7	0010	15	0110

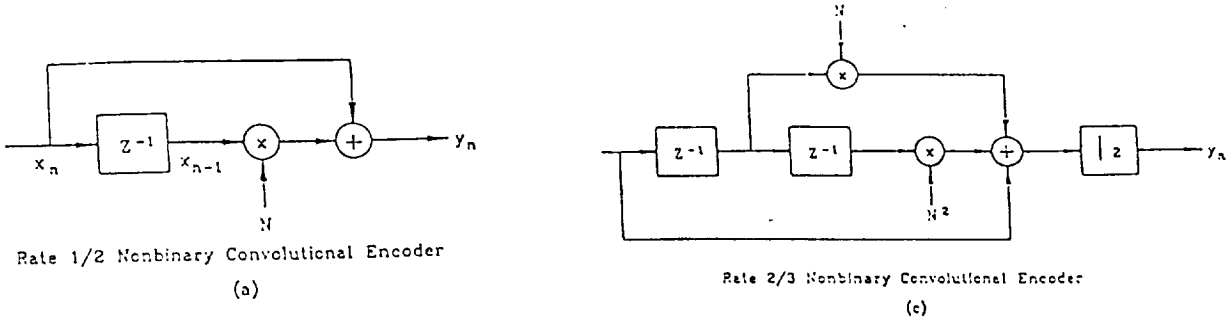

Figure 1. Proposed Nonbinary Convolutional Encoders.

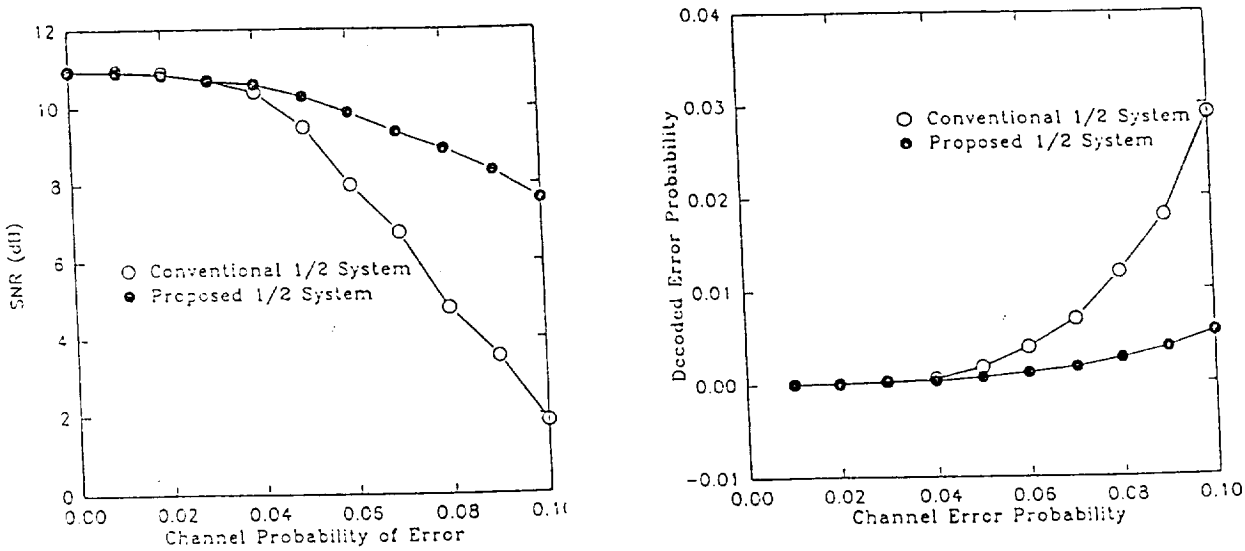

Figure 2. Performance Comparison of rate 1/2 Systems.

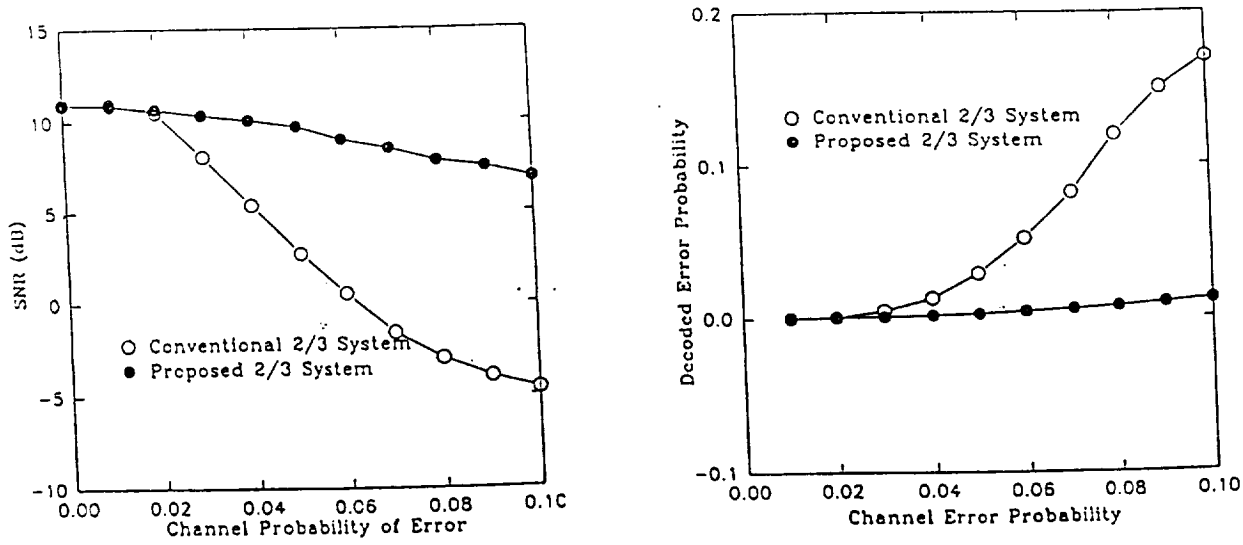

Figure 3. Performance Comparison of rate 2/3 Systems.

Appendix 2- Item 5